Disabling equational theories in unification for cryptographic protocol analysis through tagging

Sreekanth Malladi

Dakota State University,
Madison, SD - 57042, USA

malladis@pluto.dsu.edu

April 13, 2010

1 Introduction

Most of the research on protocol security in the past two decades has been conducted assuming a free message algebra. However, operators such as Exclusive-OR (XOR) possess algebraic properties. There were instances when a protocol was secure in the free algebra, but insecure in the presence of equational theories induced by such operators [10]. Hence, it is important to conduct protocol analysis with careful consideration of equational theories.

Unification is an important part of symbolic protocol analysis that is affected by equational theories. If we can disable them, i.e., if we can construct protocol messages such that unification in the presence of equational theories implies the same unification in their absence, then it is a good step in simplifying protocol analysis in the presence of theories.

This is the point we consider in this paper. We formulate a new tagging scheme for protocol messages that essentially disables disjoint equational theories[1]. As a consequence of this result, we could recently achieve the following, immensely useful result for protocols involving the XOR operator that possesses the ACUN theory:

> Under a certain tagging scheme, if a protocol is secure under a free algebra, then it is secure in the presence of the ACUN theory.

We provide a full formal proof of this result in [9]. This result essentially disables the ACUN theory from having any effect on protocol security. Further, it allows us to lift many existing results obtained under a free algebra. For instance, the classical "small-system" decidability result by Lowe in his pioneer work [8] states,

> "If there is no attack on a small system of a protocol (with exactly one agent playing each role of the protocol), leading to a breach of secrecy, then there is no attack on any larger system leading to a breach of secrecy."

Although Lowe has achieved this result in a free term algebra, we can tag protocols in our scheme, and use our main result of [9] to conclude that the small-system result is valid even under the ACUN theory, since no new attacks are enabled. We can similarly recover many existing results achieved under a free algebra, such as simplifying transformations for protocols [7], preventing type-flaw and multi-protocol attacks [6, 5].

[1] Disjoint theories are those where the equations in the theories do not share operators.

Submitted to:
UNIF 2010

Such a similar result is possible under other theories such as ACU∪Inverse and ACU∪Idempotence as well. However, while a crucial component of [9] is disabling equational unification (which is the only point of this paper), the protocol analysis framework used in [9] is from [3], which is tailored only to the ACUN theory. To achieve a similar result under other theories, we would have to use the result of the current paper in suitable protocol models such as [4]. This is a topic of current research.

It is very important to note that our result does <u>not</u> consider equations of the form $[a,b]\oplus[c,d]=[a\oplus c, b\oplus d]$, which would hold when the operator $\oplus$ is homomorphic. The reason is that, we use the algorithm by Baader & Schulz for combined theory unification [2] to achieve our result. The algorithm cannot handle such equations that use operators in disjoint theories (the above equation uses pairing, which is a free operator and the XOR operator). However, some implementations could lead to such equations, and we consider it an important direction of future research to include them.

2 Term Algebra

We will assume the existence of a basic, indivisible set of terms called variables and constants denoted as *Vars* and *Constants* respectively. We define a set of operators, $Ops = StdOps \cup eqop$, where, $StdOps = \{sequence, penc, senc, pk, sh\}$. We use some syntactic sugar in using some of these operators:
$sequence(t_1,\ldots,t_n)=[t_1,\ldots,t_n]$, $penc(t,k)=[t]^{\rightarrow}_k$, $senc(t,k)=[t]^{\leftrightarrow}_k$, $eqop(t_1,\ldots,t_n)=t_1\oplus\ldots\oplus t_n$.

Note that, although we use the symbol $\oplus$ for *eqop* that is conventionally used for XOR, here we treat it as a general operator that has some equational theory.

The term algebra is the infinite set, *Terms*, where $Vars\cup Constants\subset Terms$ and $(\forall t_1,\ldots,t_n\in Terms; f\in Ops)(f(t_1,\ldots,t_n)\in Terms)$. We will define two relations, *subterm* and *interm* denoted $\sqsubset$ and $\Subset$ respectively on terms such that:

- $t\sqsubset t'$ iff $t=t'$ or $t'=f(t_1,\ldots,t_n)$ where $f\in Ops$ and $t\sqsubset t''$ for some $t''\in\{t_1,\ldots,t_n\}$.
- $t\Subset t'$ iff $(\exists t_1,\ldots,t_m; i\in\{1,\ldots,m\})((t_1\oplus\ldots\oplus t_m=t')\wedge(t_i=t))$.
- $SubTerms(T)=\{t\mid(\exists t'\in T)(t\sqsubset t')\}$.

Interms are also subterms, but subterms are not necessarily interms. For instance, $[1,a]$ is both an interm and a subterm of $[1,a]\oplus[2,b]\oplus[3,[n_b]_k]$, but n_b is only a subterm in it, not an interm.

Definition 1. [Equation and Theory] *An* equation *is a tuple* $\langle term,\ term\rangle$. *We write* $t_1=_e t_2$ *if* $e=\langle t_1,\ t_2\rangle$ *is an equation. A* theory *is a set of equations. If Th is a theory, we write* $t=_{Th}t'$, *if there exist a finite sequence of equations* $e_1,\ldots,e_n\in Th$ *such that,* $t=_{e_1}t_1$, $t_1=_{e_2}t_2$, …, $t_{n-1}=_{e_n}t'$.

The theory STD for *StdOps* is a set of equations between syntactically equal terms:
$\{\langle[t_1,\ldots,t_n],\ [t_1,\ldots,t_n]\rangle, \langle[t]^{\leftrightarrow}_k,\ [t]^{\leftrightarrow}_k\rangle, \langle[t]^{\rightarrow}_k,\ [t]^{\rightarrow}_k\rangle, \langle pk(t),\ pk(t)\rangle, \langle sh(t_1,t_2),\ sh(t_1,t_2)\rangle\}$.

The theory EQTH has equations solely with the *eqop* ($\oplus$) operator. For our main result, we will consider the ACUN theory as EQTH, but in principle, this can be any set of equations where *StdOps* are not used. There can also be multiple operators in EQTH:
$\{\langle t_1\oplus(t_2\oplus t_3),\ (t_1\oplus t_2)\oplus t_3\rangle, \langle t_1\oplus t_2,\ t_2\oplus t_1\rangle, \langle t\oplus 0,\ t\rangle, \langle t\oplus t,\ 0\rangle\}$.
We also define FEQOP, which is a theory in which the $\oplus$ operator is free:
$\mathsf{FEQOP}=\{\langle t_1\oplus\ldots\oplus t_n,\ t_1\oplus\ldots\oplus t_n\rangle\}$.

Definition 2. [Operators]
Let $Operators(Th)$ *denote all the operators used to form the equations in the theory Th*[2]*:*

[2] We use an underscore (_) in a formula, when the value in it doesn't affect the truthness of the formula.

$$Operators(Th) = \left\{ op \mid (\exists e \in Th; t_1, t_2, t \in Terms) \left(\begin{array}{c} ((t \sqsubset t_1) \vee (t \sqsubset t_2)) \wedge \\ (e = \langle t_1,\ t_2 \rangle) \wedge (t = op(\_, \ldots, \_)) \end{array} \right) \right\}.$$

Theories Th_1 and Th_2 are disjoint *if* $Operators(Th_1) \cap Operators(Th_2) = \{\}$.

We will say that a term t is *pure* wrt the theory Th, if all of its subterms are made only from $Operators(Th)$: $\mathsf{pure}(t, Th) \Leftrightarrow (\forall op(\_, \ldots, \_) \sqsubset t)(op \in Operators(Th))$.

We will now consider equational unification. We will abbreviate "Unification Algorithm" to UA and "Unification Problem" to UP:

Definition 3. [Unification Problem, Unifier, Unification Algorithm]

A Th-UP *is a tuple of terms* $\langle m,\ t \rangle$ *denoted* $m \stackrel{?}{\approx}_{Th} t$*, where* m *and* t *are pure wrt* Th*. If* Th *is a theory, a set of* Th-UPs, Γ, *is* Th-Unifiable *with a set of substitutions* σ *called a* Th-Unifier, *if* $(\forall m \stackrel{?}{\approx}_{Th} t \in \Gamma)(m\sigma =_{Th} t\sigma)$*. A* Th-Unifier σ *is a* most general Th-Unifier, *for a set of* Th-UPs Γ, *if every other* Th*-Unifier* ρ *for* Γ *is such that,* $\rho = \sigma\rho$*. A* complete Th-UA *returns all possible most general* Th*-Unifiers for any set of* Th*-UPs.*

UAs for two disjoint theories Th_1 and Th_2, may be combined to output the unifiers for a set of $(Th_1 \cup Th_2)$-UPs using Baader & Schulz Combination Algorithm (BSCA) [2]. We give a more detailed explanation in Appendix A, using an example UP for the interested reader.

BSCA first takes as input, a set of $(Th_1 \cup Th_2)$-UPs, say Γ, and applies some transformations on them to derive $\Gamma_{5.1}$ and $\Gamma_{5.2}$ that are sets of Th_1-UPs and Th_2-UPs respectively. It then combines the unifiers for $\Gamma_{5.1}$ and $\Gamma_{5.2}$ obtained using Th_1-UA and Th_2-UA respectively (see Appendix A, Def. 5) to form the unifier(s) for Γ. Further, if all UPs in $\Gamma_{5.1}$ and $\Gamma_{5.2}$ are Th_1-Unifiable and Th_2-Unifiable respectively, then Γ is $(Th_1 \cup Th_2)$-Unifiable.

It has been proven in [2] that the combined unifier obtained is a complete $(Th_1 \cup Th_2)$-UA for any $(Th_1 \cup Th_2)$-UP if Th_1-UA and Th_2-UA are complete and if Th_1 and Th_2 are disjoint.

3 DNUT - Disabling Non-Unifiability of Terms

We now state our main requirement on terms, namely DNUT.

Definition 4 (DNUT)**.** *A set of terms* T *is* DNUT-Satisfying *or* DNUT-Satisfying(T) *iff:*

1. *No two interms of an eqop term are* STD-Unifiable[3]*:*

$$(\forall t \in SubTerms(T); n \in \mathbb{N}) \left(\begin{array}{c} (t = t_1 \oplus \ldots \oplus t_n) \wedge (n > 1) \wedge \\ (\forall i, j \in \{1, \ldots, n\})((i \neq j) \Rightarrow (t_i \not\approx_{\mathsf{STD}} t_j)) \end{array} \right).$$

2. *No interm of an eqop term is* STD-Unifiable *with an interm of any other eqop term:*

$$(\forall t, t' \in SubTerms(T)) \left((\exists t_1, t_1')((t_1 \Subset t) \wedge (t_1' \Subset t') \Rightarrow (t_1 \not\approx_{\mathsf{STD}} t_1')) \right).$$

3. *The Unity element is not a part of any eqop term:*

$$(\forall t_1 \oplus \ldots \oplus t_n \in SubTerms(T); n \in \mathbb{N}) \left((\nexists i \in \{1, \ldots, n\})(t_i = 0) \right).$$

[3] $\mathbb{N}$ is the set of natural numbers.

The first requirement of DNUT can be satisfied by ensuring that every term in the set $\{t_1,\ldots,t_n\}$ is a pair that starts with a distinct constant, if $t_1 \oplus \ldots \oplus t_n$ is a subterm of T. For instance, consider, $A \oplus N_B \oplus [N_A]^{\rightarrow}_{pk(B)} \oplus [N_B]^{\leftrightarrow}_{K}$. This can be changed to $[1,A] \oplus [2,N_B] \oplus [3,[N_A]^{\rightarrow}_{pk(B)}] \oplus [4,[N_B]^{\leftrightarrow}_{K}]$ so that, no two terms in the set $\{[1,A],[2,N_B],[3,[N_A]^{\rightarrow}_{pk(B)}],[4,[N_B]^{\leftrightarrow}_{K}]\}$ are STD-Unifiable.

To explain the second requirement of DNUT, consider another term, $B \oplus N_A \oplus [N_B]^{\rightarrow}_{pk(A)} \oplus [N_A]^{\leftrightarrow}_{N_B}$. We can introduce tags in this term as well, similar to the previous term, as $[1,B] \oplus [2,N_A] \oplus [3,[N_B]^{\rightarrow}_{pk(A)}] \oplus [4,[N_A]^{\leftrightarrow}_{N_B}]$, so as to satisfy the first requirement. However, this would violate the second requirement, since interms in both might be unifiable. For instance, $[1,A]$ in the first term is STD-Unifiable with $[1,B]$ in the second. To avoid this, we can range the interms in the first *eqop* term from 1.1 to 1.4, and the second from 2.1 to 2.4. So the terms are now, $[1.1,A] \oplus [1.2,N_B] \oplus [1.3,[N_A]^{\rightarrow}_{pk(B)}] \oplus [1.4,[N_B]^{\leftrightarrow}_{K}]$ and $[2.1,B] \oplus [2.2,N_A] \oplus [2.3,[N_B]^{\rightarrow}_{pk(A)}] \oplus [2.4,[N_A]^{\leftrightarrow}_{N_B}]$. Obviously, they satisfy the third requirement as well.

Below we give a protocol that has multiple XOR terms, to illustrate how DNUT may be satisfied in protocols where there might be many complex and nested terms:

Original protocol		**Changed to satisfy DNUT**
$A \rightarrow B:$	A,B	A,B
$B \rightarrow A:$	$[N_B,B] \oplus [N_B,A]^{\rightarrow}_{pk(A)}$	$[2.1,N_B,B] \oplus [2.2,[N_B,A]^{\rightarrow}_{pk(A)}]$
$A \rightarrow B:$	$A \oplus N_B \oplus [A \oplus N_B]^{\rightarrow}_{pk(B)} \oplus$ $[N_A]^{\leftrightarrow}_{N_B}$	$[3.1,A] \oplus [3.2,N_B] \oplus [3.3,[[3.3.1,A] \oplus$ $[3.3.2,N_B]]^{\rightarrow}_{pk(B)}] \oplus [3.4,[N_A]^{\leftrightarrow}_{N_B}]$
$A \rightarrow B:$	$[N_A \oplus N_B,A,B]^{\rightarrow}_{pk(A)} \oplus$ $[N_A \oplus A, N_B \oplus B]^{\leftrightarrow}_{N_A \oplus N_B}$	$[4.1,[[4.1.1,N_A] \oplus [4.1.2,N_B],A,B]^{\rightarrow}_{pk(A)}] \oplus$ $[4.2,[[4.2.1,N_A] \oplus [4.2.2,A],\ [4.3.1,N_B] \oplus$ $[4.3.2,B]]^{\leftrightarrow}_{[4.4.1,N_A] \oplus [4.4.2,N_B]}]$

4 Main Result

We will now prove that, if DNUT is followed in a set of terms, the effects of equational theories are totally disabled.

Theorem 1. *Let T be a set of terms that are DNUT-Satisfying. Then, if two non-variables are unifiable in the* $(\mathsf{STD} \cup \mathsf{EQTH})$ *theory, then they are also unifiable in the* $(\mathsf{STD} \cup \mathsf{FEQOP})$ *theory:*

$$\textit{DNUT-Satisfying}(T) \Rightarrow (\forall m,t \in T)((m,t \notin \textit{Vars}) \wedge (m \approx_{(\mathsf{STD} \cup \mathsf{EQTH})} t) \Rightarrow (m \approx_{(\mathsf{STD} \cup \mathsf{FEQOP})} t)).$$

Proof. Suppose $\{\langle m,\ t\rangle\} = \Gamma$.

From BSCA, for m and t to be $(\mathsf{STD} \cup \mathsf{EQTH})$-Unifiable, every $\langle m_1,\ t_1\rangle \in \Gamma_{5.1}$ should be STD-Unifiable and every $\langle m_1,\ t_1\rangle \in \Gamma_{5.2}$ should be EQTH-Unifiable:

$$(\forall \langle m_1,\ t_1\rangle \in \Gamma_{5.1})(m_1 \approx_{\mathsf{STD}} t_1) \wedge (\forall \langle m_1,\ t_1\rangle \in \Gamma_{5.2})(m_1 \approx_{\mathsf{EQTH}} t_1). \tag{1}$$

Suppose $\langle m_1,\ t_1\rangle \in \Gamma_{5.2}$. From BSCA, we have that, for $\langle m_1,\ t_1\rangle$ to be EQTH-Unifiable, for every interm x of m, there should exist a term y as an interm of m_1 or t_1 such that, x and y are STD-Unifiable (unless x is the Unity element):

$$(\forall x)\left(\ (x \Subset m_1) \wedge (x \neq 0) \Rightarrow (\exists y)(((y \Subset m_1) \vee (y \Subset t_1)) \wedge (x \approx_{\mathsf{STD}} y))\ \right). \tag{2}$$

Now from DNUT Condition 3, no *eqop* term has the Unity element as an interm. From DNUT Condition 1, interms *within* an *eqop* term should not be STD-Unifiable. Hence, y cannot be an interm

of m_1. Similarly, from DNUT Condition 2, interms *between* two different *eqop* terms should not be STD-Unifiable as well. Hence, y cannot be an interm of t_1 either.

The only other way for m_1 and t_1 to be EQTH-Unifiable is that m_1 must be equal to t_1, in which case they are both EQTH-Unifiable and FEQOP-Unifiable.

Thus in general, every $\langle m_1,\ t_1\rangle$ belonging to $\Gamma_{5.2}$ is FEQOP-Unifiable. Further, from (1), every $\langle m_1,\ t_1\rangle$ belonging to $\Gamma_{5.1}$ is STD-Unifiable.

Hence, $\langle m,\ t\rangle$ is $(\mathsf{STD}\cup\mathsf{FEQOP})$-Unifiable.

□

5 Conclusion

In this paper, we showed that tagging messages that were constructed with operators possessing algebraic properties, disables the equational theories induced by those properties.

Tags specified in DNUT basically disable cancellation of terms entirely, both inside a term or between different terms. For ACUIdem and ACUInverse, no change is required at all in DNUT. For other theories that are disjoint with the standard theory, we can use similar tagging to disable cancellation, and disable the theories. In the presentation and the full paper, we will explain those details and also the impact of the main result on symbolic protocol analysis.

The main result easily falls apart under homomorphic encryption (HE). For instance, the UP $[1,A] \stackrel{?}{\approx} [3,a]\oplus[6,b]\oplus[4,C]$ has DNUT-Satisfying terms. It is unifiable under $\mathsf{STD}\cup\mathsf{HE}$ with $\{a/A, b/C\}$ as the unifier, if binary encoding is used for the tags 3, 4 and 6, since $[3,a]\oplus[6,b]\oplus[4,C] = [011\oplus 110\oplus 100, a\oplus b\oplus C]$ under HE, which is equal to $[1,a]$ if $C=b$. But it is not under $\mathsf{STD}\cup\mathsf{FEQOP}$.

It seems that extending the result under non-disjoint theories such as STD and HE will be quite challenging. Although BSCA cannot be used, I conjecture that new unification algorithms such as [1] might be useful in this pursuit. I look forward to discussions with the workshop participants toward further work in this direction.

References

[1] S. Anantharaman, H. Lin, C. Lynch, P. Narendran & M. Rusinowitch (2009): *Unification Modulo Homomorphic Encryption*. In: *FroCos*, pp. 100–116.

[2] F. Baader & K. U. Schulz (1996): *Unification in the Union of Disjoint Equational Theories: Combining decision procedures*. *J. of Symbolic Computation* 21, pp. 211–243.

[3] Y. Chevalier (2004): *A simple constraint solving procedure for protocols with exclusive-or*. *Presented at Unif 2004 workshop* Available at http://www.lsv.ens-cachan.fr/unif/past/unif04/program.html.

[4] S. Delaune, P. Lafourcade, D. Lugiez & R. Treinen (2008): *Symbolic protocol analysis for monoidal equational theories*. *Inf. Comput.* 206(2-4), pp. 312–351.

[5] J. D. Guttman & F. J. Thayer (2000): *Protocol Independence through Disjoint Encryption*. *13th IEEE Computer Security Foundations Workshop* , pp. 24–34.

[6] J. Heather, G. Lowe & S. Schneider (2000): *How to prevent type flaw attacks on security protocols*. In: *Proc. 13th Computer Security Foundations Workshop*, IEEE Computer Society Press, pp. 255–268.

[7] M. L. Hui & G. Lowe (1999): *Safe Simplifying transformations for security protocols*. In: *12th Computer Security Foundations Workshop Proceedings*, IEEE Computer Society Press, pp. 32–43.

[8] G. Lowe (1999): *Towards a completeness result for model checking of security protocols*. *Journal of Computer Security* 7(2-3), pp. 89–146.

[9] S. Malladi (2010): *Disabling* ACUN *theory for cryptographic protocol analysis through tagging*. Technical Paper Number DSU-BIS-IA-2010C, Available at `http://www.homepages.dsu.edu/malladis/`.

[10] P. Y. A. Ryan & S. A. Schneider (1998): *An Attack on a Recursive Authentication Protocol. A Cautionary Tale. Inf. Process. Lett.* 65(1), pp. 7–10.

A Bader & Schulz Combined theory unification algorithm

In this section, we will describe Bader & Schulz's combination algorithm [2] (abbreviated to BSCA) that combines unification algorithms for two disjoint theories.

We will use the following $(\mathsf{STD} \cup \mathsf{ACUN})$-UP as our running example:

$$[1, n_a]^{\rightarrow}_{pk(B)} \approx^{?}_{\mathsf{STD} \cup \mathsf{ACUN}} [1, N_B]^{\rightarrow}_{pk(a)} \oplus [2, A] \oplus [2, b].$$

Step 1 (Purify terms). BSCA first "purifies" the given set of $(Th = Th_1 \cup Th_2)$-unification problems, Γ, into a new set of problems Γ_1 through the introduction of some new variables such that, all the terms are "pure" wrt Th_1 or Th_2, but not both.

If our running example was Γ, then, the set of problems in Γ_1 are $W \approx^{?}_{\mathsf{STD}} [1, n_a]_{pk(B)}$, $X \approx^{?}_{\mathsf{STD}} [1, N_B]_{pk(a)}, Y \approx^{?}_{\mathsf{STD}} [2, A]$, $Z \approx^{?}_{\mathsf{STD}} [2, b]$, and $W \approx^{?}_{\mathsf{ACUN}} X \oplus Y \oplus Z$, where W, X, Y, Z are obviously new variables that did not exist in Γ.

Step 2. (Purify problems) Next, BSCA purifies the unification problems such that every problem in the set has both terms belonging to the same theory. For our example problem, this step can be skipped since all the problems in Γ_1 already have both their terms purely from the same theory (STD or ACUN)).

Step 3. (Variable identification) Next, BSCA partitions variables in Γ_2 into a partition *VarIdP* such that each variable in Γ_2 is replaced with a representative from the same equivalence class in *VarIdP*. The result is Γ_3.

In our example problem, one set of values for *VarIdP* can be $\{\{A\}, \{B\}, \{N_B\}, \{W\}, \{X\}, \{Y, Z\}\}$.

Step 4. (Split the problem) The next step of BSCA is to split Γ_3 into two sets of problems such that each set $\Gamma_{4.i}$ has every problem with terms from the same theory, Th_i ($i \in \{1, 2\}$).

Following this in our example,

$$\Gamma_{4.1} = \{\langle W,\ [1, n_a]_{pk(B)}\rangle, \langle X,\ [1, N_B]_{pk(a)}\rangle, \langle Y,\ [2, A]\rangle, \langle Z,\ [2, b]\rangle\},$$

and

$$\Gamma_{4.2} = \{\langle W,\ X \oplus Y \oplus Y\rangle\}.$$

Step 5. (Solve systems) The penultimate step of BSCA is to partition all the variables in Γ_3 into a size of two: Let $p = \{V_1, V_2\}$ is a partition of $Vars(\Gamma_3)$. Then, the earlier problems ($\Gamma_{4.1}$, $\Gamma_{4.2}$) are further split such that all the variables in one set of the partition are replaced with new constants in one of the set of problems and vice-versa in the other.

In our sample problem, we can form $\{V_1, V_2\}$ as $\{Vars(\Gamma_3), \{\}\}$. i.e., we choose that all the variables in problems of $\Gamma_{5.2}$ be replaced with new constants. This is required to find the unifier for the problem (this is the partition that will successfully find a unifier).

So $\Gamma_{5.1}$ stays the same as $\Gamma_{4.1}$, but $\Gamma_{5.2}$ is changed to $\Gamma_{5.2} = \Gamma_{4.2}\beta = \{\langle W,\ X \oplus Y \oplus Y\rangle\}\beta = \{\langle w,\ x \oplus y \oplus y\rangle\}$. i.e., $\beta = \{w/W, x/X, y/Y\}$, where, w, x, y are constants, which obviously did not appear in $\Gamma_{5.1}$.

Step 6. (Combine unifiers) The final step of BSCA is to combine the unifiers obtained in Step 5 for $\Gamma_{5.1}$ and $\Gamma_{5.2}$:

Definition 5. [Combined Unifier]

Let Γ be a Th-UP where $(Th_1 \cup Th_2) = Th$. Let $\sigma_i \in A_{Th_i}(\Gamma_{5.i}$, $i \in \{1,2\}$ and let $V_i = Vars(\Gamma_{5.i})$, $i \in \{1,2\}$.

Suppose '$<$' is a linear order on $Vars(\Gamma)$ such that $Y < X$ if X is not a subterm of an instantiation of Y:

$$(\forall X, Y \in Vars(\Gamma))((Y < X) \Rightarrow (\nexists \sigma)(X \sqsubset Y\sigma)).$$

Let $\mathsf{least}(X, T, <)$ be defined as the minimal element of set T, when ordered linearly by the relation '$<$'. i.e.,

$$\mathsf{least}(X, T, <) \Leftrightarrow (\forall Y \in T)((Y \neq X) \Rightarrow (X < Y)).$$

Then, the combined UA for Γ, namely $A_{Th_1 \cup Th_2}$, is defined such that,

$$A_{Th_1 \cup Th_2}(\Gamma) = \{\sigma \mid (\exists \sigma_1, \sigma_2)((\sigma = \sigma_1 \odot \sigma_2) \wedge (\sigma_1 \in A_{Th_1}(\Gamma_{5.1})) \wedge (\sigma_2 \in A_{Th_2}(\Gamma_{5.2})))\}.$$

where, if $\sigma = \sigma_1 \odot \sigma_2$, then,

- *The substitution in σ for the least variable in V_1 and V_2 is from σ_1 and σ_2 respectively:*

  $(\forall i \in \{1,2\})((X \in V_i) \wedge \mathsf{least}(X, Vars(\Gamma), <) \Rightarrow (X\sigma = X\sigma_i))$*; and*

- *For all other variables X, where each Y with $Y < X$ has a substitution already defined, define $X\sigma = X\sigma_i\sigma$ $(i \in \{1,2\})$:*

  $(\forall i \in \{1,2\})((\forall X \in V_i)((\forall Y)((Y < X) \wedge (\exists Z)(Z/Y \in \sigma))) \Rightarrow (X\sigma = X\sigma_i\sigma))$.